Mikhail Razakov

C++ / Systems Engineer | LLVM · x86-64 · Algorithms

razakov.mikhail@outlook.com
t.me/Micodiy
github.com/Misha1302

I build C++/LLVM and low-level systems components: conservative IR transformations, x86-64 code generation, and memory-dependence analysis. Exact affine counting, bounded algorithms, and independent oracles constrain what the tools may claim.

Professional and project evidence

ISP RAS: SharpChecker symbolic computation · MCST: LLVM 22/C++23.

x86-64 pipeline

SSA/CFG validation, liveness/ interference, linear scan, seeded simulated annealing, SysV codegen, and differential execution.

Conservative optimization

LLVM Global-IV: APInt affine evolution, transitive effects, explicit legality, and 29 verifier/idempotence/execution regressions.

EXPERIENCE

October 2026 — present

ISP RAS — SharpChecker development

- Work on symbolic computation in SharpChecker, a C#/.NET static-analysis tool.

July 1 — August 31, 2026

MCST — Compiler Engineering Intern

- Implemented an LLVM LICM pass in C++; transformation legality is derived from loop structure, memory accesses, side effects, and speculative-safety constraints.
- Built a Python harness comparing execution before/after optimization and checking explicit transformation / no-transformation cases; also implemented RPO, Dijkstra, Dinic, and Tarjan SCC in C++23.

PROJECTS

LLVM Interprocedural Global IV Optimization

A conservative LLVM 22 prototype/MVP with an actual IR transform; 29 positive/negative regression cases cover verifier validity, idempotence, and before/after execution.

x86-64 Compiler Backend & Register Allocation

A reference interpreter and differential execution check generated-code semantics; the allocator assignment contract is verified independently.

PS-form Memory Dependence Analyzer

Exact affine iteration-pair counting and modular/GCD disjointness proofs; yes/no cross-checked with a small-domain oracle and metamorphic tests, unresolved cases remain maybe.

SKILLS

C++ / toolchain

C++23, C17, LLVM 22, CMake, Linux, clang-tidy, ASan/UBSan

Algorithms / analysis

graph algorithms, CFG, dominance, SSA, SCC, dataflow, memory dependence

Machine-level systems

x86-64 SysV, code generation, liveness/interference, register allocation, generated-code analysis

Verification

exact/reference oracles, differential/metamorphic testing, isolated execution, reproducible counterexamples

EDUCATION

HSE University — Software Engineering

RECOGNITION

MCST LLVM/C++ selection

Organizer correspondence: only 5.0/5.0 score and only solution passing all 104/104 tests among 49 submissions.

LangDev'26 — Build the Language, Then Make the Abstractions Disappear

Delivered the UniversalToolchain architecture talk at LangDev on 9 October 2026.

Moscow / remote